R1.4: Anforderungsaufnahme an IoT Tests

für CoAP, MQTT und OPC-UA

Gefördert durch:

Bundesministerium für Wirtschaft und Energie

aufgrund eines Beschlusses des Deutschen Bundestages

Version 0.1 Datum: 15.08.2019
Version 1.0 Datum: 20.11.2019
Version 1.1 Datum: 21.11.2019
Version 1.2 Datum: 22.11.2019

Autoren:

Frank-Walter Jäkel (Ed)	Fraunhofer IPK
Tobias Wolff	Fraunhofer IPK
Alexander Kaiser	Relayr
Sascha Hackel	Fraunhofer FOKUS
Axel Rennoch	Fraunhofer FOKUS
Andre Wardaschka	DEKRA

1 Einleitung

Der Report fasst die Anforderungen zu IoT-Tests zusammen, welche im Laufe des Projektes zusätzlich zu den Anforderungen zu Projektbeginn (Report 1.1 und 1.2) gesammelt wurden. Die Anforderungen basieren auf Erfahrungen der letzten 2,5 Jahre, welche teilweise in der Form von Szenarien in dem internen Report „R3.1: Spezifikation der IoT-Testszenarien im Projekt“ dokumentiert wurden. Ein Teil der Anforderungen wurde für die Weiterentwicklungen in IoT-T berücksichtigt. Dabei wurden insbesondere Anforderungen aus der Industrie zu OPC-UA und Performanz von IoT Protokollen detailliert.

Dazu gehören:

- Sicherheit und Performanz von MQTT und CoAP,
- Interoperabilität im Zusammenhang mit OPC-UA
- Verwendung unterschiedlicher OPC-UA SDKs,
- Modellierung und Nutzung von Anwendungsszenarien zum automatisierten Test bezüglich Interoperabilität von OPC-UA Anwendungen.

Andere Anforderungen wurden ebenfalls von Anwendern und der Industrie als relevant eingestuft können aber in der verbleibenden Projektbearbeitungszeit nicht berücksichtigt werden. Hierzu gehören die Verwaltung von Geräten und Interaktion zur Laufzeit. Diese werden soweit sie von den Anwendern detailliert wurden im Bericht als Bedarfe für zukünftige Erweiterungen festgehalten.

2 Anforderungen

2.1 Interoperabilität und OPC-UA

Zu Interoperabilität existieren unterschiedlichste Definitionen. Eine mehr generische Definition, die von vielen Organisationen wie ETSI, ISO und auch Interop-Vlab (Enterprise Interoperability) herangezogen und meist weiter detailliert wird, ist folgende:

„The ability of two or more systems or components to exchange data and use the exchanged information“.

Interoperabilität beinhaltet sowohl technische als auch geschäftliche oder organisatorische Aspekte und benötigt ein gemeinsames Verständnis der zu verarbeitenden Daten, womit auch die Semantik ins Spiel kommt. Die Semantik wird dabei im Wesentlichen durch Ontologien abgebildet.

Interoperabilität ist eine unabdingbare Voraussetzung für eine erfolgreiche Digitalisierung in einem vernetzten Ökosystem wie in Unternehmensnetzwerken. Sie kommt insbesondere bei der Vernetzung von Anlagen untereinander und mit Unternehmensanwendungen - z.B. mit Supply Chain Management (SCM), Produkt Lebenszyklus Management (PLM), Manufacturing Execution Systems (MES) zum Tragen.

Aktuell wird OPC-UA als Framework für die Vernetzung von Maschinen und Unternehmens-anwendungen von großen Unternehmen stark forciert. OPC-UA [1] bietet eine gemeinsame Infrastruktur in Bezug auf die Kombination von Clients und Servern. Es hat ein Konzept der Verwendung von Kommunikationsprotokollen wie TCP oder MQTT [2], wobei die spezifische Implementierung vom verwendeten Kommunikationsprotokoll abhängt. OPC-UA beinhaltet verschiedene Modi für

Sicherheitsniveaus und hat ein Konzept zum Aufbau von Informationsmodellen sowie von funktionalen Semantiken mit den Conformance Units [3]. Daher bietet OPC-UA einen guten Ausgangspunkt für die Bereitstellung von Interoperabilität. Aktuell arbeiten Standardisierungsorganisationen und verwandte Verbände wie der VDMA an bereichsspezifischen Informationsmodellen (Companion Specs.). Die spezifische Semantik von Parametern und Einheiten ist jedoch auch mit OPC-UA noch eine Herausforderung.

In das Projekt IoT-T wurde OPC-UA über den Anwendungsfall der modularen Shopfloor IT vom IoT-T Partner AUDI zusammen mit dem Fraunhofer IPK eingebracht. Nach der Klärung bezüglich der Nutzung von IoT in der Industrie und der Industrieanforderung, dass OPC-UA unterstützt werden soll (siehe R1.2), wurde diese Anforderung von allen Partnern aufgenommen und weiterverfolgt.

OPC-UA bietet mit einer einheitlichen Architektur die Voraussetzungen für Interoperabilität. Für eine konkrete Implementierung benötigt OPC-UA aber eine passende Definition von Informationsmodellen (Companion Specs.) und deren Konformität untereinander.

Damit sind sowohl Compliance Tests als auch die Überprüfung der Interoperabilität bezogen auf die Informationsmodelle erforderlich. Diese sind eine Vorbedingung für ein sicheres digitales Plug-and-Produce. OPC-UA stellt für die Compianance Tests das UA Compliance Test Tool (UACTT) zur Verfügung, welche sich auf die Definition von Compliance Units abstützen. Diese können durch weitere Anforderungen beispielsweise von spezifischen Companion Specs. erweitert werden.

Im Projekt IoT-T wird eine Prüfung der Informationsmodelle zur Verfügung gestellt, welche sich aus den industriellen Anforderungen ableitet. Dazu wurden unterschiedliche generelle Testfälle definiert, welche sukzessive in die von ETSI favorisierte „Test Description Language“ (TDL) überführt werden. Hierdurch wird eine einheitliche, formale und nachvollziehbare Beschreibung der Testfälle erreicht.

Im Laufe des Projekts haben sich folgende Anforderungsszenarien konkretisiert:

- Zusammenspiel unterschiedlicher Informationsmodelle,
- Updates der Unternehmensanwendungen und der Verbindung zu OPC-UA Schnittstellen,
- Nutzen der „standardisierten“ Informationsmodelle,
- Digital Plug-and-Produce.

Es lassen sich 3 Aspekte erkennen:

1. Absicherung der unterliegenden Protokolle z.B. MQTT bezüglich Sicherheit, Compliance und Performance,
2. Absicherung der Compliance bezüglich des OPC-UA Frameworks,
3. Absicherung der Interoperabilität der genutzten Informationsmodelle.

Die Punkte 1 und 2 sind Voraussetzung für den Punkt 3 (Abbildung 1). Dabei wird der Punkt 1 durch die Testware des IoT-T Projektes bezüglich MQTT abgedeckt. Der Punkt 2 nutzt die Testwerkzeuge der OPC Fundation und der Punkt 3 wird durch die Werkzeuge zur Validierung von OPC-UA Schnittstellen im IoT-T Projekt erarbeitet. Hier können auch die Werkzeuge der OPC Fundation genutzt werden, diese erfordern aber die Programmierung der spezifischen Tests. Damit entsteht die Anforderung Überprüfungen der Informationsmodelle auch ohne zusätzlichen Programmieraufwand zu ermöglichen.

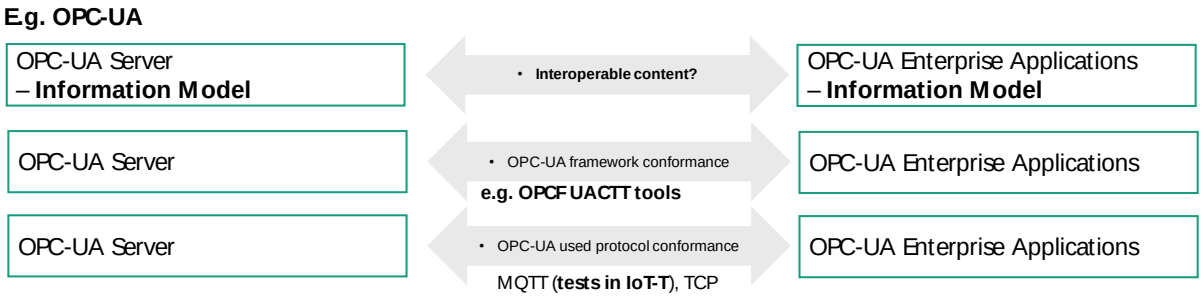

Abbildung 1: Interoperabilität von OPC-UA Implementierungen

2.2 Abhängigkeit vom OPC-UA SDK

OPC-UA SDKs (Software Development Kits) stehen von unterschiedlichen Quellen in unterschiedlichen Programmiersprachen zur Verfügung. Es existieren Open Source und freie Softwarelösungen, als auch kostenpflichtige Angebote von Softwarelieferanten. Dabei ist der Grad der Umsetzung der OPC-UA Spezifikation unterschiedlich und auch der Bezug zur OPC-UA Version wichtig. Beispielsweise sind zurzeit die neueren Publisher-Subscriber Ansätze nicht immer verfügbar. Daher ist auch die Validierungssoftware von dem jeweiligen SDK abhängig. Daraus leitet sich die Forderung ab, hier möglichst flexibel zu sein und sowohl das SDK, als auch die verwendete OPC-UA Framework Implementierung austauschen zu können.

Ziele sind:

- Für ein konkretes Informationsmodell die Validierung mit OPC-UA Framework Implementierung durchzuführen, welche auch im Anwendungsfall genutzt wird.
- Gegeneinander Laufenlassen von unterschiedlichen OPC-UA Framework Implementierungen, welche von unterschiedlichen Herstellern bereitgestellt werden.
- Nutzung unterschiedlicher Sprachen und Validierung der OPC-UA Framework Implementierungen in diesen Sprachen wie zum Beispiel SDKs in C, C++, JAVA.

Insbesondere aus der Industrie wird hier eine Einheitlichkeit und bessere Absicherung gefordert.

Im Projekt werden hier Versuche durchgeführt bezüglich unterschiedlicher OPC-UA JAVA SDKs (Client) und deren Austausch sowie eines OPC-UA C++ SDK (Server).

2.3 Sicherheit (Security)

Die Sicherheit, insbesondere die Security, gewinnt mit zunehmender Digitalisierung einen immer höheren Stellenwert und stellt bei der Öffnung von Systemen zum Internet immense Risiken dar. Während die Themen *Security* und *Privacy* im herkömmlichen Gebrauch des Internet bereits eine hohe Sensibilisierung erreicht haben, gestaltet sich die Schutzbedürftigkeit von Maschinen, die über das Internet kommunizieren wesentlich komplexer. Aber nicht nur die Tatsache, dass die Angriffsfläche durch die bloße Anzahl der Kommunikationsteilnehmer und deren Quervernetzung rapide ansteigt, erhöht das Risiko für Sicherheitsvorfälle. Durch die Vernetzung und Digitalisierung von Industrieanlagen (und Aktuatoren im Allgemeinen) rücken auch die Themen Security und Safety immer näher zusammen. Während Vorfälle mit Bezug auf die Cybersecurity in der Vergangenheit meistens die digitale

Privatsphäre von Nutzern verletzten, wird es in Zukunft auch immer wichtiger sein *Security* im Gesamtkontext zusammen mit *Safety* und *Privacy* zu betrachten.

Das Thema „Security“ bildet im Zuge der Digitalisierung und damit möglichen Anbindungen der Produktion an das Internet auch organisatorische Herausforderungen. Verantwortliche für die Fertigung werden gegebenenfalls auch für IT Sicherheitsaspekte herangezogen und die IT Verantwortlichen müssen sich mit Maschinenschnittstellen befassen. Ohne klare Verantwortlichkeiten wird sich die Sicherheit vor Cyberangriffen nur schwer umsetzen lassen. Diese Anforderung geht über das IoT-T Projekt hinaus, wurde aber in den Diskussionen mit Industriepartnern sichtbar; beispielsweise durch Aussagen der Fertigungsleitung: „Wozu müssen wir uns mit Sicherheit befassen, solange die Fertigung läuft, machen wir keine Änderungen oder Updates“.

2.4 Performanz

Ein weiterer Aspekt bei der Betrachtung der Qualität im Bereich des IoT ist die Performanz. Nachdem Konformität und Interoperabilität sichergestellt wurde, folgt die Betrachtung der Performanz. Hauptziel ist die Überprüfung der Machbarkeit. Darüber hinaus können am Ende des Performanztestens konkrete Konfigurationen zur Optimierung für MQTT-Lösungen herausgearbeitet werden.

Methodisch werden nacheinander verschiedene Typen des Performanztestens durchgeführt. Dabei handelt es sich um folgende:

1. Lasttest
2. Langzeittest
3. Stresstest
4. Konfigurationstest
5. Szenariotest

Es wird mit normaler Last für das zu testende System gestartet und im Verlauf des Testprozesses weiter gesteigert. Nach einem Langzeittest, bei dem das System unter hoher Last über einen langen Zeitraum gefordert wird, folgt der Stresstest. Hierbei findet eine Überladung mit Netzwerkverkehr statt und überprüft insbesondere das Verhalten in Bezug auf die Resilienz. Durch dieses Vorgehen werden die technische Kapazität und Grenzen des Systems ausgelotet. Die letzten beiden Aktivitäten sind Konfigurationstests und Szenariotests. In ersterer wird das Verhalten in Bezug auf Performanz für unterschiedliche Konfigurationen evaluiert. Daraus können Empfehlungen und ein optimales Vorgehen abgeleitet werden. Für das Szenariotesten wird "realistischer" Netzwerkverkehr abgeleitet und gegen das System gespielt. Hierbei wird das Vertrauen in die Lösung gestärkt und somit evaluiert, ob das System einsatzbereit ist.

2.5 Vernetzung

Die meisten Tests und Validierungen untersuchen einzelne Sender/Empfänger Kommunikationen. Die Vernetzung unterschiedlicher Systeme führt zu weiteren Herausforderungen. Beispiel hierfür ist die Vernetzung unterschiedlicher Anlagen in der Produktion. Dieses kann durch Kombinationen von Clients und Servern in OPC-UA mit einer übergreifenden Steuerung erfolgen. Ein entsprechendes Szenario bildet die modulare Shopfloor IT. Unterschiedliche Anlagen werden in diesem Fall über eine

Ausführungskomponente (Execution Engine) miteinander lose verknüpft (Abbildung 2). Die Reihenfolge und die erforderlichen Ausführungsservices werden dabei durch ein leicht anpassbares Ablaufmodell definiert.

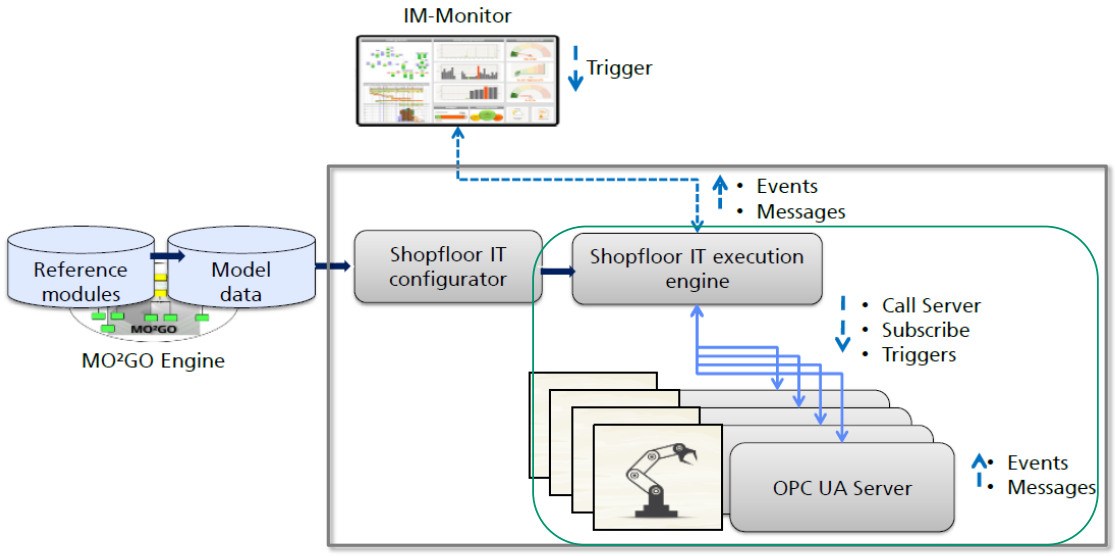

Abbildung 2: Modellbasierte, modulare Shopfloor IT mit OPC-UA [4]

Daraus leiten sich weitere Anforderungen ab, wie

1. Abhängigkeiten zwischen Daten und Prozessen, welche, zeitlich und von ihrer Reihenfolge abhängig, über die Vernetzung entstehen: Ein Datum muss erst vorhanden sein, bevor es weiterverarbeitet wird.
2. Lücken zwischen der Ausführung von Diensten auf den Anlagen etwa durch fehlende Anlagenverbindungen.

Diese Anforderungen sollten in zukünftigen Arbeiten berücksichtigt werden.

2.6 Robustheit

Die Robustheit einer Software oder eines Systems bezeichnet die Fähigkeit wie stabil ein System auf Fehlbedienungen, unerwartete Eingaben oder äußere Einflüsse reagiert. Mangelnde Robustheit kann dazu führen, dass Angreifer ein System von außen manipulieren könnten ein unerwartetes Verhalten auszulösen. Dies kann im einfachsten Fall zum Beispiel dazu führen, dass ein (Teil-) System abstürzt und für das Gesamtsystem, wie etwa eine digitale Produktionsstraße, nicht mehr verfügbar ist.

Eine solche Verletzung der Verfügbar kann zu Denial-of-Service (DoS) Angriffen führen, die unter Umständen eine gesamte Produktionsstraße lahmlegen könnten. In hochvernetzten Umgebungen mit unterschiedlichen digitalen Systemen können einzelne angreifbare Teilsysteme oft auch als Einfallstore für weitere Angriffsszenarien ausgenutzt werden. Daher ist die Robustheit jedes einzelnen Systems auch entscheidend für die Sicherheit und Robustheit des gesamten Systemverbundes.

Um die Robustheit der Implementierungen von Kommunikationsprotokollen wie MQTT und CoAP zu überprüfen, wird im Rahmen des Projektes der Ansatz des Fuzzy Testings in Kombination mit den vorhandenen Konformitätsprüfungen untersucht und erprobt.

3 Anforderungen an die Software

3.1 Usability

Die Benutzerfreundlichkeit (Usability) sollte stets im Kontext einer Kosten-Nutzen-Analyse betrachtet werden. In gewissen Fällen ist eine gewisse Komplexität unvermeidbar, der resultierende Nutzen muss diese Komplexität rechtfertigen.

Auf der anderen Seite sind durchaus auch relativ simple Fälle denkbar, in denen die Testsoftware hilfreich sein kann, wenn diese dem Benutzer mit einfachen Mitteln einen Mehrwert bieten. Hierzu zählen vor allem kleinere bis mittelgroße Aufgaben mit hohem Wiederholungscharakter wie Netzwerk- und Portscans, Regressionstests oder auch die Validierung von Informationsmodellen nach einem Update oder einer Wartung.

Insbesondere sind folgende Anforderungen für Test- und Validierungswerkzeuge, die schnell und leicht von möglichst vielen Nutzern eingesetzt werden sollen, relevant:

- Einfache und aufwandsarme Anwendung der Testsoftware, da sonst die Nutzung nicht erfolgt und damit auch kein positiver Effekt erreicht wird.
- Die Softwarelösungen sollen einerseits leicht über Intranet/Cloud Zugänge erreichbar, aber für auch die lokale Nutzung in einer Fabrik, oder bei einem Lieferanten verfügbar sein.
- Eine Rückverfolgbarkeit und Vergleichbarkeit von Tests sollte angeboten werden.
- Ergebnisse müssen leicht verständlich und leicht verfügbar sein, dazu sollten sie auch Informationen zur Erklärung der Ergebnisse beinhalten.

Eine wichtige Anforderung insbesondere bis Ende 2019 ist die Konsolidierung der bestehenden Lösungen, sodass diese von Nutzern leicht eingesetzt und von Entwicklern weiterentwickelt werden können.

3.2 Fehlertoleranz und Verlässlichkeit

Test- und Prüfwerkzeuge sollten nicht wegen inkorrekter Eingaben stehenbleiben oder abstürzen, sondern entsprechende Meldungen liefern, welche vom Nutzer weiterverarbeitet werden können. Insbesondere wurden folgende Anforderungen definiert:

- Vertrauen in die Ergebnisse bezüglich der verwendeten SDKs,
- Keine Abstürze wegen Fehlern in den Eingabedaten oder durch Bedienungsfehler,
- Erklärung und Nachvollziehbarkeit der Ergebnisse.

Neben der eigentlichen Anwendung muss auch die Testsoftware robust gestaltet werden. Hier wird durch Tests, und auch durch den Einsatz unterschiedlicher Werkzeuge zur Prüfungen von Protokollen und Schnittstellen, eine bessere Absicherung möglich. Bezüglich OPC-UA können zum einen zusätzlich

die OPC Testwerkzeuge der OPC Fundation zur Absicherung, zum anderen unterschiedliche OPC-UA SDKs für die Tests genutzt werden.

4 Standards

4.1 Überblick

Im Rahmen der Anforderungen wurden sowohl unterschiedliche Standardisierungsansätze aus ISO, IEEE, OMG und ETSI betracht, als auch Industriestandards. Das Projekt hat sich dabei auf folgende Ansätze fokussiert:

- OPC-UA ist ein de-facto Standard und wird zurzeit massiv von der Industrie in Deutschland, aber auch international vorangetrieben, auch wenn der Einsatz in der Industrie noch überschaubar ist. Andere Rahmenwerke wie DDS (Data Distribution Service) können zukünftig aus den USA auch nach Deutschland kommen. Diese sollten entsprechend mituntersucht werden. Konzepte aus DDS wie Publisher-Subscriber finden sich in neueren Versionen von OPC-UA auch wieder. Im Rahmen des IoT-T Projekts wird sich aufgrund der deutschen Ausrichtung im Wesentlichen mit OPC-UA befasst.
- MQTT ist ein leichtgewichtiges Datentransferprotokoll welches im Bereich IoT und M2M zunehmend eingesetzt wird. Für die wachsende Beliebtheit in diesen Bereichen gibt es mehrere Gründe. Einer davon ist die Schlichtheit und Einfachheit des Protokolls. Verglichen zu anderen Protokollen kommt MQTT mit relativ wenig Features und einer überschaubaren Spezifikation aus. Das spiegelt sich auch in der großen Anzahl der vorhandenen Open-Source Implementierungen aus. Auf der anderen Seite bietet MQTT durch das Publish-Subscribe-Konzept eine hohe Flexibilität und Einsatzmöglichkeiten in unterschiedlichen Szenarien.
  Insbesondere für IoT und M2M bietet MQTT einen entscheidenden Vorteil: ein Großteil der gesamten Komplexität des Protokolls wird von dem Broker abgewickelt. Der Broker als zentrales Bindeglied zwischen allen beteiligten Kommunikationsteilnehmern übernimmt neben den eigentlichen Protokoll-Features eine besondere Rolle in Bezug auf Security und Performanz. Der Broker ist somit ein möglicher *Single-Point-of-Failure* in der MQTT-Kommunikation und erfordert daher eine gesonderte Absicherung in Bezug auf Konformität, Sicherheit und Performanz, aber auch auf die korrekte Konfiguration und Inbetriebnahme. Auf der anderen Seite sind die MQTT-Clients verhältnismäßig schlank und anspruchslos. Durch flexible Topics auf dem Broker können Clients zwar untereinander kommunizieren, jedoch nicht direkt in die Sessions der anderen Clients (bei korrekter Implementierung des Brokers) eingreifen.
- CoAP ist ein Kommunikationsprotokoll, welches das Client-Server Paradigma umsetzt und auf UDP basiert. Es gewann im Zusammenhang mit dem IoT an Bedeutung. Der Fokus des

Protokolls liegt insbesondere in der Übertragung telemetrischer Sensordaten in instabilen Umgebungen, sodass die Vorteile des verbindungslosen UDP-Protokolls zum Tragen kommen. Das CoAP-Protokoll wird betrachtet, aber aufgrund seiner derzeitig eher geringen Nutzung nur konzeptionell miteinbezogen.

4.2 Compliance

Die „Compliance“ von Protokollen und Rahmenwerken ist eine Grundvoraussetzung für andere Anforderungen, wie Sicherheit und Interoperabilität. Für MQTT wurden im Projekt bereits entsprechende Tests prototypisch umgesetzt und sogenannte „Test Purposes“ in TDL umgesetzt und bei ETSI eingebracht. Für OPC-UA existieren schon entsprechende Testwerkzeuge und sogenannte Compliance Units der OPC Foundation. Weitere Informationen bzgl. OPC-UA und „Compliance“ finden sich auch unter Kapitel 2.1.

4.3 Informationsmodell OPC-UA

Die Informationsmodelle in OPC-UA spielen eine entscheidende Rolle bei der Interoperabilität von Schnittstellen zwischen OPC-UA Clients und OPC UA Servern. Große Unternehmen haben hier die Chance Unternehmensstandards zu entwickeln und diese am Markt für ihre Anlagen durchzusetzen. Kleine und mittlere Unternehmen brauchen hingegen verlässliche Standards und auch große Unternehmen sind interessiert, ihre Kosten bezüglich der Standardisierung von Informationsmodellen zu senken. Daher gibt es eine Reihe von Bestrebungen Informationsmodelle zu standardisieren.

Das VDMA hat hierzu in Deutschland verschiedene Projekte und erste Standards, siehe hierzu https://opcua.vdma.org/. Prototypen der entsprechenden Informationsmodelle sind in GITHUB verfügbar und werden in IoT-T im Zuge der Entwicklung des CPS ValidationAdapters (CPS-VA) und des CPS Emulators (CPS-E) genutzt und verifiziert.

Ein Beispiel für die Nutzung der Standardisierten Informationsmodelle findet sich unter folgendem Link: https://dashboard.umati.app/. Erste Standards sind beispielsweise EUROMAP77 (Anwender, Anlagenanbieter und MES Anbieter) https://opcua.vdma.org/viewer/-/v2article/render/27038001 und OPC-UA Companion Specifications für Robotik und Industrielle Bildverarbeitung https://industrie40.vdma.org/viewer/-/v2article/render/26418188.

Offen ist vorerst noch die Interoperabilität zwischen den Informationsmodellen, woran zurzeit bereits gearbeitet wird. Ein Test auf Konformität zu den Standards ist ebenfalls über das OPC UA Compliance Test Tool (UACTT) der OPC Foundation möglich. Ein schneller und einfacher Test über den im IoT-T entwickelten CPS-VA ist insbesondere für angepasste Informationsmodelle erforderlich, um Abweichungen schnell identifizieren zu können.

5 Fallstricke

5.1 IT Infrastruktur und Organisation

Eine Anforderung, die insbesondere die Sicherheit betrifft, aber auch Performanz und Interoperabilität, liegt in der Sensibilisierung für die Informationstechnik und Schnittstellenanforderungen in der Fertigung. In der Vergangenheit fühlten sich die Fertigungsleiter wenig verantwortlich für die IT, und die IT Abteilung nicht für die Maschinen in der Fertigung. Jetzt haben die Maschinen digitale Schnittstellen und können prinzipiell als Objekt im Internet sichtbar werden. Typische Aussagen und Vorgehensweisen in der Fertigung müssen daher hinterfragt werden, wie beispielsweise:

- Die Aussage, es sei „sicher da lokales Netz“, welche gerne in der Produktion genutzt wird, kann sich schnell durch Fernwartung oder IoT Geräte, wie fernsteuerbare Lampen, ändern.
- Die Ansätze, solange die Fertigung korrekt und effizient liefe, seien „keine Updates nötig“ und auch nicht erlaubt, ist in Hinblick auf die Dynamik in der Softwareentwicklung schwer einzuhalten - insbesondere bei Sicherheitsupdates und Schnittstellen zu Unternehmensanwendungen wie ERP und MES.

Damit werden einfache Test- und Validierungswerkzeuge notwendig um beispielsweise schnell nach einem Update auch von in der IT ungeschultem Personal die Schnittstellen überprüfen zu können.

5.2 „Wir sind interoperable, denn wir folgen dem Standard“

Standardisierung ist eine wichtige Grundlage für die Interoperabilität - auch zwischen Unternehmen. Leider existiert eine Vielzahl von Standards und quasi-Standards. Daher ist die Nutzung *eines* Standardprotokolls und *eines* standardisierten Informationsmodells nicht ausreichend. Beispiele hierfür sind folgende

- OPC-UA versus DDS,
- TCP versus MQTT versus CoAP,
- OPC-UA (mit TCP), OPC-UA (mit MQTT),
- Unterschiedliche standardisierte Informationsmodelle.

Damit bleibt die Überprüfung einer spezifischen Implementierung zur Absicherung der Interoperabilität notwendig. Das gilt auch entsprechend für Aussagen wie, „Systeme können sich austauschen mit unserer Architektur“.

6 Testwerkzeuge und Frameworks

Im Laufe des Projekts wurden eine Reihe von Testwerkzeugen und Frameworks identifiziert und analysiert. Die folgende Auswahl an Werkzeugen wurden identifiziert und teilweise direkt in den Lösungen des Projekts genutzt. Hierbei wurde insbesondere auch auf Open Source Werkzeuge geachtet. Um Industrieanforderungen besser prüfen zu können, wurden aber auch kommerzielle Lösungen wie „Softing“ berücksichtigt:

Protokoll / Framework	Funktion	Werkzeug / SDK
MQTT	MQTT Funktionstests / Konformität und Interoperabilität	• Eclipse Interop Testing Plan und Eclipse Paho: MQTT Conformance/Interoperabilit Testing • Synopsys MQTT Client Test Suite • Synopsys MQTT Server Test Suite • SmartBear MQTT Test Steps
MQTT	MQTT Packet Manipulation / Robustness Testing / Fuzzing / Security	• MQTT-PWN • Scapy MQTT • mqtt_fuzz • mqtt_fuzzing • mqtt-packet-fuzzy • imqtt - An interactive MQTT packet manipulation shell based in IPython • Synopsys MQTT Client Test Suite • Synopsys MQTT Server Test Suite
MQTT	MQTT Benchmarking / Performance	• mqtt-benchmark • mqtt-bench • mqtt-benchmark (nodejs) • mqtt-malaria • Gatling-MQTT • mqtt-jmeter • Bevywise IoT-Simulator • MIMIC MQTT Simulator
MQTT and CoAP	MQTT Benchmarking / Performance	• CoAP jmeter Plugin
CoAP	CoAP Funktionstests / Konformität und Interoperabilität	• Californium Framework • CoAPthon
CoAP	CoAP Packet Manipulation / Robustness Testing / Fuzzing / Security	• Scapy CoAP • FuzzCoAP
OPC-UA	Entwicklungswerkzeuge	• OPC Foundation UA JAVA Legacy

		• MILO • Softing
OPC-UA	Visualisierung von und Navigieren in Informationsmodellen eines laufenden Servers	• UA Expert
OPC-UA	CoAP Funktionstests / Konformität und Interoperabilität	• OPC UA Compliance Test Tool (UACTT)

Weitere OPC-UA Implementierungen sind Bestandteil von OPC-UA Servern, welche im Laufe des Projektes zu Testzwecken genutzt wurden. Hier wurde deutlich, dass die Implementierung OPC-UA Abweichungen haben kann, welche sowohl die Tests, als auch die spätere Anwendung beeinflussen kann. Beispielsweise hat der UA Expert in einigen Tests die Verbindung zum Server verweigert, obwohl der Server lief und der CPS ValidationAdapter erfolgreich ausgeführt werden konnte.

Compliance Tests zur besseren Absicherung bieten sich an, wie sie von der OPC Foundation vorgeschlagen werden: https://opcfoundation.org/developer-tools/certification-test-tools/opc-ua-compliance-test-tool-uactt.

7 Industrieller Anwendung

Unternehmen entwickelten ihre eigenen Infrastrukturen, um von der Verknüpfung von

- Fertigungseinrichtungen mit Unternehmensanwendungen (z.B. BDE, ERP) zu profitieren,
- die Fernwartung von Anlagen zu ermöglichen,
- Prozessabläufe zu harmonisieren,
- eine möglichst automatische Steuerung der Produktion (Selbststeuerung) zu erlauben, oder
- ein erweitertes Monitoring der Fertigung bereitzustellen.

Daher wurde das digitale Plug-and-Produce oder die virtuelle Inbetriebnahme der Steuerungssoftware wichtig. Ein industrielles IoT erlaubt eine dezentrale und flexiblere Verknüpfung von Fertigungsanlagen als die bisherigen Technologien mit Feldbussen [5]. Die Einrichtung und Ergänzung durch zusätzliche Anlagen sollte ohne Programmieraufwand erfolgen können. Dies erfordert Interoperabilität hinsichtlich der Architektur, der Protokolle, der Sicherheit und des verwendeten Informationsmodells inklusive der Semantik. Gleichzeitig wird eine hohe Sicherheit bei der Datenübertragung erwartet und ein effektiver Schutz gegen Angriffe, ohne dabei Einbußen in der Performanz verzeichnen zu müssen.

8 Zusammenfassung und Ausblick

Der Report gibt einen Überblick über Anforderungen und Ansätzen zu IoT Tests und entsprechenden Tools unter Berücksichtigung der im IoT-T Projekt verfolgten Protokolle und Frameworks (MQTT, CoAP, OPC-UA) und über die Anforderungen, welche über den Reports R1.2 aus 2017 hinausgehen. Ein Fokus

liegt dabei auf der Überprüfung der industriellen Anwendung von Geräten und Anlagen, welche über die Betriebliche IT Infrastruktur mit dem Internet verbunden werden können.

Aktuelle Bearbeitung bis Dezember 2019 sind ausgerichtet an

- Konsolidierung der erarbeiteten Werkzeuge,
- Test der Performanz für MQTT und CoAP,
- Standardisierung der Test-Purposes in ETSI,
- Automatische Ausführung von Anwendungsszenarien zur OPC-UA Prüfung,
- Anwendungsszenarien für die Absicherung der Interoperabilität von Anlagen,
- Demonstratoren,
- Veröffentlichung der Test- und Validierungswerkzeuge.

9 Referenzen

1 OPC-UA. https://opcfoundation.org/about/opc-technologies/opc-ua/ - letzter Zugriff 31.07.2019

2 MQTT Version 3.1.1 OASIS Standard 29 October 2014. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html - letzter Zugriff 31.07.2019.

3 OPC Foundation. https://opcfoundation.org/certification/overview-benefits/ - letzter Zugriff 31.07.2019.

4 Jaekel, F.-W.; Torka, J.; Eppelein, M.; Schliephack, W.; Knothe, T. Model based, modular configuration of cyber physical systems for the information management on shop-floor. International Workshop on Enterprise Integration, Interoperability and Networking (EI2N) 12, 2017, Rhodes. OTM Workshops 2017: 16-25.

5 feldbusse.de. https://www.feldbusse.de/profinet/profinet.shtml - letzter Zugriff 31.07.2019.